华中科技大学计算机科学与技术学院 2020~2021 第一学期

“C++程序设计”考试试卷（A 卷）

考试方式 开卷 考试日期 **2020-11-14** 考试时长 **150 分钟**

专业班级 _____ 学　　号 _____ 姓　　名 _____

题号	一	二	三	四	五	六	总分	核对人
分值	15	20	20	15	15	15	100	
得分								

分　数	
评卷人	

一、单选题：请从 4 个选项中选择一个最合适的选项作为答案（15 分：每小题 3 分）。

解答内容不得超过装订线

1. 关于定义“struct A{int x; mutable int y;}const a={1,3};”，如下叙述哪个__B___正确:
   A. a.x 可被赋值，a.y 不可被赋值　　B. a.x 不可被赋值，a.y 可被赋值
   C. a.x 和 a.y 均不可被赋值　　D. a.x 和 a.y 均可被赋值
2. 关于 union 定义的类的叙述___A__正确:
   A. 既不能是基类也不能是派生类　　B.不能是基类，但可以是派生类
   C. 可以是基类，但不能是派生类　　D.既可以是基类，也可以是派生类
3. 对于说明“int &f( ); int &&g( );”及其函数调用 f( )和 g( )，如下 4 个叙述___D__正确:
   A. 调用 f( )和调用 g( )均不可被赋值　　B.调用 f( )不可被赋值，调用 g( )可被赋值
   C. 调用 f( )和调用 g( )均可被赋值　　D.调用 f( )可被赋值，调用 g( )不可被赋值
4. 对于定义“struct A{ int f( ) ;}”，关于 int 前面是否可用 static 和 virtual，如下的叙述__C___错误:
   A. 可以只使用 static　　B. 可以只使用 virtual
   C. 必须同时使用 static 和 virtual　　D. 都不对(错)
5. 对于定义“char *const &f( );”，如下哪个语句是错误的___A__:
   A. f( )=(char*) "abcd";　　B. *f( )= 'A';
   C. char *p=f( );　　D. *f( )= "ABC"[1];

分　数	
评卷人	

二、在最多使用单重作用域例如 A::x 的前提下，在空白处填写以下各类可被访问的成员及其访问权限（20 分：根据正确回答的成员个数按比例计算给分）。

```
class A{            //类 A 的可访问成员：
    int a;          //私有成员：a;
protected:          //保护成员：b, c;
    int b, c;       //公有成员：d, e;
```

```
public:
    int d, e;
};
class B: protected A {//类 B 的可访问成员：
    int a;              //私有成员：a;
protected:              //私有成员：
    int b, f;           //保护成员：b, f;  A::(b, c, d, e);
    using A::d;         //保护成员：
public:                 //公有成员：e, g;
    int e, g;           //公有成员：
};
struct C: A {           //类 C 的可访问成员：
    int a;              //私有成员：
protected:              //私有成员：
    int b, f;           //保护成员：b, f;  A::(b, c);
public:                 //保护成员：
    int e, g;           //公有成员：a, e, g;  A::(d, e);
    using A::d;         //公有成员：
};
struct D: B, C {        //类 D 的可访问成员：
    int a;              //私有成员：
protected:              //私有成员：
    int b, f;           //保护成员：b, f;  B::(b, f;  A::(b, c, d, e));  C::(b, f;  A::(b, c))
public:                 //保护成员：
    int e, g;           //公有成员：a, e, g;  B::(e, g);  C::(a, e, g;  A::(d, e))
};                      //公有成员：
```

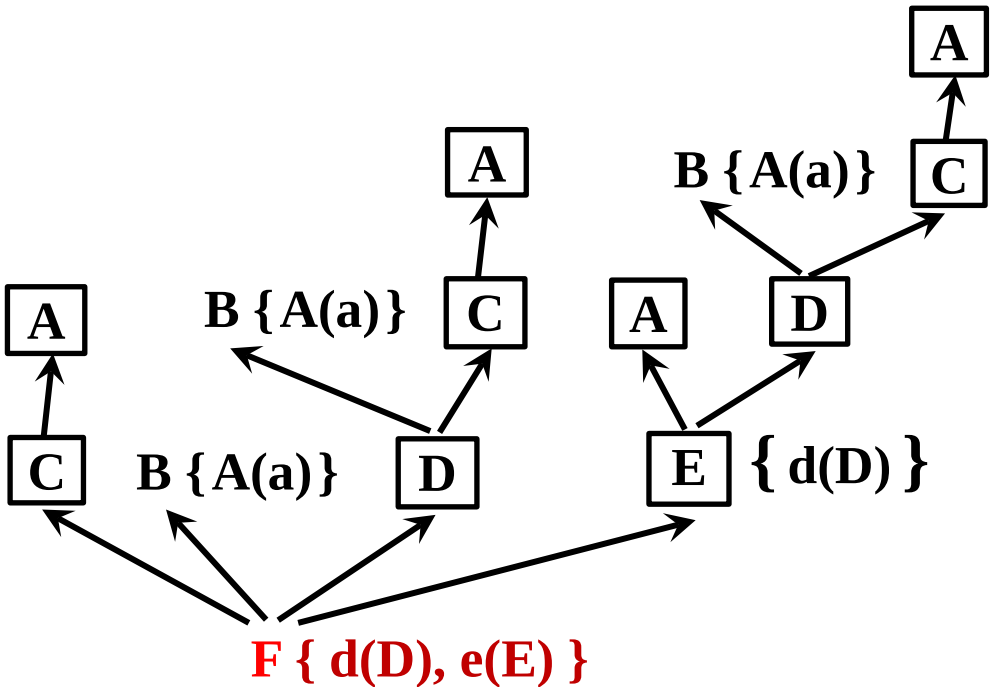
分　数	
评卷人	

三、回答 main 中每行语句的输出结果（20 分：前四个语句的输出每个 3 分，后两个语句的输出每个 4 分）。

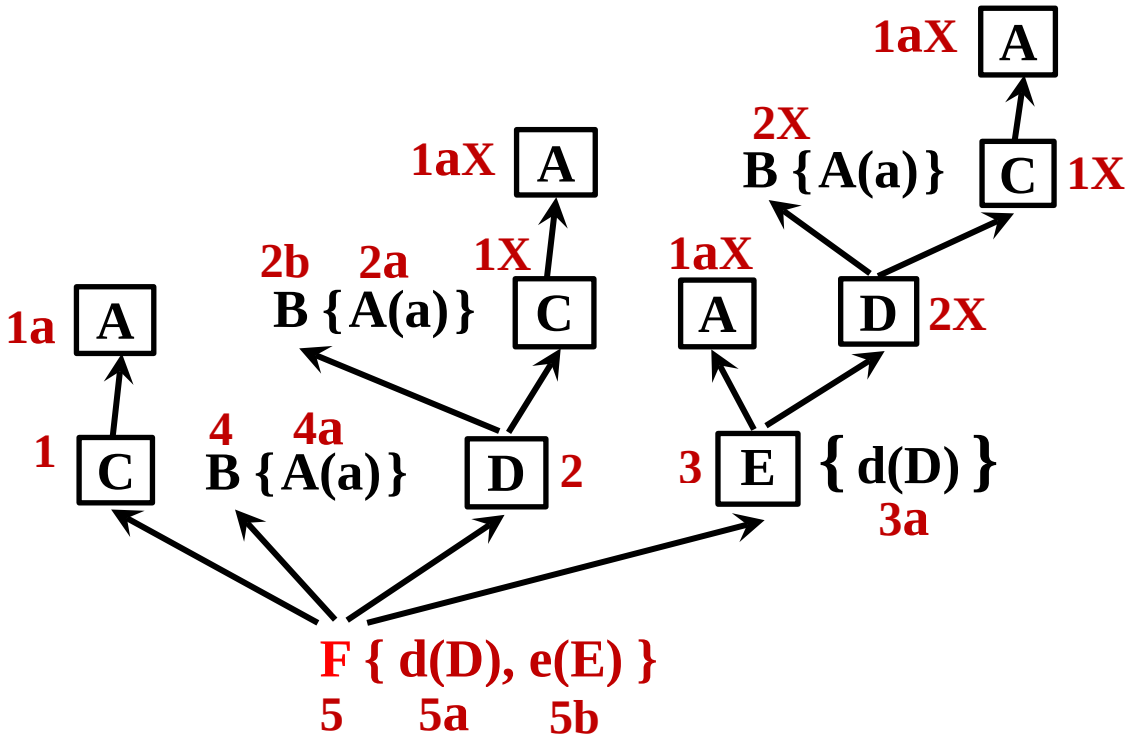
```
#include <iostream>
using namespace std;
struct A { A( ) { cout << 'A'; } };
struct B { A a; B( ) { cout << 'B'; } };
struct C : virtual A { C( ) { cout << 'C'; } };
struct D : B, virtual C { D( ) { cout << 'D'; } };
struct E : virtual A, virtual D {
    D  d;
    E( ) : A( ) { cout << 'E'; }
};
struct F : virtual C, B, virtual D, virtual E {
    D  d;  E  e;
```

```
    F( ) { cout << 'F'; }
};
void main( ) {
    A   a;   cout << '\n';          //输出=A
    B   b;   cout << '\n';          //输出=AB
    C   c;   cout << '\n';          //输出=AC
    D   d;   cout << '\n';          //输出=ACABD
    E   e;   cout << '\n';          //输出=ACABDACABDE
    F   f;   cout << '\n';          //输出=ACABDACABDEABACABDACABDEF
}
```

F f 分析如下：



解答内容不得超过装订线



D: ACABD

E: ACABD ACABD E

F: AC ABD ACABDE AB ACABD ACABDE F

分　数	
评卷人	

四、综合分析并指出以下程序中下划线位置可能出现的语法错误及其原因 (共 15 分：每错约 1 分)。

```
class A {
      int    a;
protected:
      virtual ~A( ) { }
public:
      int& b;
      int    c;
      virtual A(*g)( )=0;            //错误1：不能用virtual定义数据成员_____
      virtual A(int x)__{ a = x; };  //错误2：构造函数不能用virtual定义_错误3：没有初始化b____
} a = (4, 3);                        //错误4：不能调用保护的析构函数_____
class B: A {
      int    d;
      using A::a;                    //错误5：私有a不可访问_____
public:
```

```
    friend int operator( )(int) { return 2; }; //错误6：（）定义为非成员函数的友元_____
    B(int x, int y, int z)____{ d = x + y + z; }; //错误7：未初始化基类_____
} b(5, 6, 7);
class C: B {
    int z;
public:
    ~C(int x) { z = x; };          //错误8：不应有参数_____
} c ;                              //错误9：无法生成无参构造函数_____
void  main( ) {
    int  A::* p = &A::b;           //错误10：无法取得引用成员的地址_____
    int  i = a.a;                  //错误11：无法访问私有成员_____
    int&& y = i ;                  //错误12：无址引用不能引用有址变量_____
    i = b.b;                       //错误13：无法访问私有的 B::b_____
    i = i + c.d;                   //错误14：不存在 c.d_____
    i = b.*p;                      //错误15：B非A的子类，不能使用实例成员指针int A::*_____
}
```

解答内容不得超过装订线

分　数	
评卷人	

五、请填入自己学号的最后一位十进制数字，计算 main 函数中变量 i 在每条赋值语句执行后的值 (共 15 分：每小题 2.5 分)。

```
int  x = 填写自己学号最后一位十进制数, y = x + 3;
struct A {
    int x = ::x + 2;
    static int &y;
public:
    operator int( )const { return x + y; }
    int& v(int& x) {
        for (int y = 1; x < 301; x ^= y, y++)
            if (x > 300) { x -= 31; y -= 2; }
        return ++x;
    }
    A& operator++( ) { ++x; ++y; return *this; }
    A(int x, int y = ::y + 3) { A::y = y; }
};
int &A::y = ::y;
void main( ) {
    A  a(2, 7), b(5);
    int  i, &j = i, A::*p = &A::x;
    i = a.y;                       //i=
    j = a.x;                       //i=
    i = a.*p;                      //i=
    i = ++a;                       //i=
```

```
    i = b.y + ::y;                  //i=
    (b.v(i) = 5) += 2;              //i=
}
```

答案：

	i = a.y	j = a.x	i = a.*p	i = ++a	i = b.y + ::y	(b.v(i) = 5) += 2
0	10	2	2	14	22	7
1	10	3	3	15	22	7
2	10	4	4	16	22	7
3	10	5	5	17	22	7
4	10	6	6	18	22	7
5	10	7	7	19	22	7
6	10	8	8	20	22	7
7	10	9	9	21	22	7
8	10	10	10	22	22	7
9	10	11	11	23	22	7

分析：

	::x	**::y, A::y**	**a.x**	**b.x**	**i**
	n	**n+3**			
a(2, 7)		**7**	**n+2**		
b(5)		**10**		**n+2**	
i = a.y					**10**
j = a.x					**n+2**
i = a.*p					**n+2**
i = ++a		**11**	**n+3**		**n+14**
i = b.y + ::y					**22**
(b.v(i)=5) += 2					**7**

分　数	
评卷人	

六、对于如下所有单词按升序排列的字典类 DCT，对其中的函数成员进行程序设计(共 15 分：每个函数 1.5 分)。

```
class DIC {
    char** const e;                 //用于存放字典的单词
    const int m;                    //能够存放的单词个数
    int r;                          //已经存放的单词个数
public:
    DIC(int m=1000);                //创建最多存 m 个单词的字典，所有 e[i]置空指针
```

```
    DIC(const DIC& d);                          //根据已知字典 d 深拷贝构造新字典
    DIC(DIC&& d)noexcept;                       //根据已知字典 d 移动构造新字典
    DIC& operator=(const DIC& d);               //深拷贝赋值运算符的重载
    DIC& operator=(DIC&& d)noexcept;            //移动赋值运算符的重载
    DIC& operator<<(const char* w);             //将单词插入字典并保持升序，若有该单词则不插
    operator int( )const noexcept;              //获得字典实际存放的单词个数
    const char* operator[ ](int i)const noexcept;  //获得下标为 i 的单词
    int operator( )(const char* w)const noexcept;  //查找单词 w 在字典中的位置
    ~DIC( )noexcept;                            //析构字典
};
```

答案：

```
#define _CRT_SECURE_NO_WARNINGS
#include <string.h>
DIC::DIC(int m) :e(new char* [m] {}), m(e ? m : 0), r(0) { };
DIC::DIC(const DIC& d) :e(new char* [d.m]{}), m(e ? d.m : 0), r(d.r) {
    if (e) throw "memory not enough!";
    for (int h = 0; h < r; h++) {
        e[h] = new char[strlen(d.e[h]) + 1];
        if (e[h] == nullptr) throw "memory not enough!";
        strcpy(e[h], d.e[h]);
    }
};
DIC::DIC(DIC&& d)noexcept : e(d.e), m(d.m), r(d.r) {
    (char**&)(d.e) = nullptr;
    (int&)(d.m) = r = 0;
}
DIC& DIC::operator=(const DIC& d) {
    if (this == &d) return *this;
    if (e) {
        for (int h = 0; h < r; h++) {
            if (e[h]) delete [ ] e[h];
        }
        delete [ ]e;
    }
    (char**&)e = new char* [d.m];
    if (e== nullptr) throw "memory not enough!";
    (int&)m = d.m;
    r = d.r;
    for (int h = 0; h <r; h++) {
        e[h] = new char[strlen(d.e[h]) + 1];
        if (e[h] == nullptr) throw "memory not enough!";
        strcpy(e[h], d.e[h]);
    }
    return *this;
}
DIC& DIC::operator=(DIC&& d)noexcept {
    if (this == &d) return *this;
```

解答内容不得超过装订线

```
    if (e) {
        for (int h = 0; h < r; h++) {
            if (e[h]) delete[ ] e[h];
        }
        delete[]e;
    }
    (char**&)e=d.e;
    (int&)m = d.m;
    r = d.r;
    (char**&)(d.e) = nullptr;
    (int&)(d.m) = r = 0;
    return *this;
}
DIC& DIC::operator<<(const char* w) {
    if (operator( )(w) != -1) return *this;
    int h;
    if (r == m) throw "memory not enough for insert!";
    for (h = 0; h < r; h++)
        if (strcmp(e[h], w) > 0) {
            for (int x = r; x > h; x--) e[x]=e[x - 1];
            e[h] = new char[strlen(w) + 1];
            if (e[h] == nullptr) throw "memory not enough!";
            strcpy(e[h], w);
            r++;
            break;
        }
    return *this;
}
DIC::operator int( ) const noexcept { return r; }
const char* DIC::operator[ ](int i)const noexcept {
    if (i<0 || i>=r) throw "subscription overflowed!";
    return e[i];
}
int DIC::operator( )(const char* w)const noexcept {
    for (int h = 0; h < r; h++)
        if (strcmp(e[h], w) == 0) return h;
    return -1;
}
DIC::~DIC( ) noexcept {
    if (e) {
        for (int h = 0; h < r; h++) {
            if (e[h]) delete[ ] e[h];
        }
        delete [ ]e;
        (char**&)e = nullptr;
        (int&)m = r = 0;
    }
}
```